Australian Government
Department of Defence
Defence Science and
Technology Organisation

An Analysis of Task Scheduling for a Generic Avionics Mission Computer

R. B. Dodd

Air Operations Division
Defence Science and Technology Organisation

DSTO-TN-0691

ABSTRACT

Task scheduling is investigated for a set of generic tasks representative of a heavily loaded mission computer application. A number of commonly used scheduling algorithms are applied to the generic task set, and a range of schedulability analysis calculations are performed. Suitability of the scheduling algorithms to the mission computer application is determined, and characteristics of different schedulability analysis techniques are established through their application to the generic task set.

Published by

DSTO Defence Science and Technology Organisation
506 Lorimer St
Fishermans Bend, Victoria 3207 Australia

Telephone: (03) 9626 7000
Fax: (03) 9626 7999

AR-013-635
April 2006

APPROVED FOR PUBLIC RELEASE

An Analysis of Task Scheduling for a Generic Avionics Mission Computer

Executive Summary

Software developed for real-time mission computing applications is typically partitioned into a number of tasks requiring independent execution. Scheduling the execution of these tasks on a multi-tasking computer requires observance of timing constraints for each and every task if tasks are to meet their real-time performance goals. The difficulty of achieving a scheduling solution increases as total processor loading increases, and task scheduling assumes a significant importance in the development of avionics mission computer software.

DSTO task DST 00/061, "Avionics Enabling Research and Development", has developed models of avionics mission systems to support analysis of mission computers and their task sets. This report documents analysis performed in support of model development, considering a number of commonly used scheduling algorithms and schedulability analysis methods. Performance of each scheduling algorithm is determined for a representative mission computer and task set, and characteristics of different analysis methods are presented.

The scheduling algorithms and schedulability analyses that are documented in this report will assist in model construction, permitting simulation and evaluation of mission computer software architectures to identify performance risks, to assess alternative architectures, and to predict the impact of changes to an architecture.

Contents

1. INTRODUCTION 1

2. TASK SET SPECIFICATION 1

3. SCHEDULABILITY ANALYSES 2
3. 1 Preemptive Fixed Priority Scheduling 2
3. 1. 1 Basic Rate-Monotonic Schedulability Test 2
3. 1. 2 Workload Analysis 3
3. 1. 3 Response Time Analysis 8
3. 2 Non-Preemptive Fixed Priority Scheduling 10
3. 2. 1 Simple Non-Preemptive Response Time Test 10
3. 2. 2 Non-Preemptive Response Time Test 12
3. 3 Asynchronous Fixed Priority Scheduling 14
3. 4 Dynamic Priority Scheduling 24
3. 5 Static Scheduling 25

4. CONCLUSIONS 27

5. REFERENCES 28

1. Introduction

Software developed for real-time applications is typically partitioned into separate tasks exhibiting logical or temporal independence. Executing these software tasks on a multi-tasking real-time computer system requires observance of timing constraints for every task, in order to meet the required levels of real-time performance. Judicious resolution of the competing demands of multiple tasks is needed if all tasks are to meet their timing constraints, and is identified as the problem of task execution scheduling.

Task scheduling can be approached as part of the software system design, resulting in a statically determined tasking schedule that is followed at execution time. This approach offers efficient run-time operation, but suffers from the need to go through a potentially costly schedule design change when any single task is changed. An alternative is to employ a dynamic scheduling algorithm to perform task scheduling at run-time, where predictable algorithm operation allows analysis to confirm acceptable performance for a given set of tasks. The disadvantages of dynamic scheduling include additional complexity and run-time overheads in the run-time kernel.

This report considers a number of commonly used approaches to task scheduling and applies them to a hypothetical mission system. A range of schedulability analysis calculations are used to demonstrate methods for confirming whether scheduling performance will be satisfactory, and some general conclusions are drawn on the strengths and weaknesses of scheduling approaches and analysis methods.

2. Task Set Specification

Locke et al [1] have developed an informal specification for a set of mission computer tasks on a hypothetical avionics mission system. This specification provides a realistic example of the type of mission computing demands encountered in currently fielded fighter aircraft with legacy 16-bit mission computer architectures. Total loading on the mission computer is very high, presenting a demanding scenario to use in the investigation of task scheduling solutions.

The mission computer task set developed by Locke et al [1] is summarised in Table 1, where timings are given in units of milliseconds. Aperiodic tasks are handled by scheduling them periodically with periods equal to their deadlines. There are no inter-task synchronisation or communication requirements, allowing all tasks to execute independently. Only a single processor is available to be used for the execution of all tasks. Tasks that were specified with maximum periods of 55 and 52 ms have had their periods rounded down to 50 ms, to simplify analysis at the expense of slightly increasing total processor utilisation. Processor utilisation is 97. 5%, compared with 95. 1% before rounding down any task periods.

Table 1. Generic avionics mission system tasks

Task Name	Computation Time (ms)	Period (ms)	Deadline (ms)
Weapon release	1	10	5
Radar tracking	2	40	
Target tracking	4	40	
Target sweetening	2		40
HOTAS bomb button	1		40
Aircraft flight data	8	50 (55)	
HUD display	6	50 (52)	
MPD tactical display	8	50 (52)	
Steering	6	80	
Weapon trajectory	7	100	
Threat response display	3		100
AUTO/CCIP toggle	1		200
Poll RWR	2	200	
Reinitiate trajectory	6		400
Periodic BIT	5	1000	400

3. Schedulability Analyses

Task scheduling solutions can be broadly categorised according to whether they are statically determined or dynamically determined using a run-time scheduling algorithm. Within the category of dynamic scheduling, prioritisation of tasks can be statically fixed or dynamically determined at run-time, and task preemption may or may not be used. The following sections demonstrate schedulability analyses for a number of variations of dynamically determined scheduling, and a single statically determined schedule is also presented.

3.1 Preemptive Fixed Priority Scheduling

Preemptive scheduling of tasks according to fixed priorities can be analysed in a number of ways if appropriate conditions are met. For the purposes of demonstrating the different schedulability analyses, variations will be made to the task set specifications where necessary to allow an analysis to be applied. Analysis is performed using the basic rate-monotonic schedulability test in §3.1.1, using processor workload demand in §3.1.2, and using task response times in §3.1.3.

3.1.1 Basic Rate-Monotonic Schedulability Test

Algorithms for preemptive scheduling of statically and dynamically prioritised computer tasks were investigated by Liu and Layland [2], who coined the term rate-monotonic scheduling to describe the assignment of fixed task priorities based on a monotonic ordering of task invocation rate. They proved the rate-monotonic scheduling algorithm was an optimal scheduling algorithm, in that no other fixed priority scheduling algorithm could schedule a task set that could not be scheduled using the rate-monotonic algorithm.

They also developed a basic schedulability test that establishes a sufficient, but not necessary, condition for confirming that a set of tasks will be successfully scheduled.

Although the generic avionics mission system tasks do not have all task deadlines equal to task periods as required by the rate-monotonic algorithm, it is not impossible for the algorithm to successfully schedule such task sets and the analysis will still be performed. For this analysis, it will be assumed that the deadline for the *Weapon release* task is 10 ms, and that the deadline for the *Periodic BIT* task is 1000 ms. The basic rate-monotonic schedulability test uses task computation times C_i and periods T_i, for each task i, to calculate total processor utilisation which must be less than the utilisation bound $U_{(n)}$ for the number of tasks n. The following inequality expresses this schedulability test, and it is applied to the tasks of the generic avionics mission system,

$$\sum_{i=1}^{n} \frac{C_i}{T_i} \leq U_{(n)} = n(2^{\frac{1}{n}} - 1),$$

$$\frac{1}{10} + \frac{2+4+2+1}{40} + \frac{8+6+8}{50} + \frac{6}{80} + \frac{7+3}{100} + \frac{1+2}{200} + \frac{6}{400} + \frac{5}{1000} \leq U_{(15)} = 15(2^{\frac{1}{15}} - 1),$$

$$0.975 \leq U_{(15)} = 0.709.$$

Since the processor utilisation of 0. 975 exceeds the utilisation bound of 0. 709, the task set is not guaranteed schedulable by the rate-monotonic algorithm. Failing this sufficient, but not necessary, condition for schedulability, further analysis is required to confirm that the task set is indeed not schedulable.

3.1.2 Workload Analysis

Lehoczky et al [3] have developed a workload analysis that determines a necessary and sufficient condition for rate-monotonic schedulability. This analysis is useful to determine task set schedulability in cases where the basic rate-monotonic schedulability test can not confirm schedulability. The analysis requires that task deadlines be equal to task periods, so it will again be assumed that the deadline for the *Weapon release* task is 10 ms, and that the deadline for the *Periodic BIT* task is 1000 ms.

Performing the analysis involves the calculation of cumulative processor workload demands $W_i(t)$, for each task i over a closed interval of time $[0,t]$, and corresponding loadings $L_i(t)$, L_i and L. These values are defined by the formulae

$$W_i(t) = \sum_{j=1}^{i} C_j \left\lceil \frac{t}{T_j} \right\rceil,$$

$$L_i(t) = \frac{W_i(t)}{t},$$
$$L_i = \min_{0<t\le T_i} L_i(t), \text{ and}$$
$$L = \max_{1\le i\le n} L_i .$$

A simplification is made to avoid minimisation over the continuous variable t, by testing at the rate-monotonic scheduling points of the piecewise monotonically decreasing $L_i(t)$, which yield local minima. These scheduling points occur at multiples of the periods of tasks of higher priority than task i, and for task i are represented by the set

$$S_i = \left\{ kT_j \,\middle|\, j = 1,\ldots,i;\ k = 1,\ldots,\left\lfloor \frac{T_i}{T_j} \right\rfloor \right\}.$$

Task i is schedulable by the rate-monotonic algorithm if and only if the loading L_i satisfies the rate-monotonic workload analysis schedulability test of

$$L_i = \min_{t\in S_i} L_i(t) \le 1,$$

and the entire task set is schedulable by the rate-monotonic algorithm if and only if

$$L = \max_{1\le i\le n} L_i \le 1.$$

To determine if the tasks of the generic avionics mission system are schedulable by the rate-monotonic algorithm, it is efficient to first determine the largest subset of tasks that can be proved to be schedulable using the basic rate-monotonic schedulability test. Starting with the highest priority task, and incrementally adding tasks until the basic rate-monotonic schedulability test fails,

$$\sum_{i=1}^{1} \frac{C_i}{T_i} = 0.1 \quad \le \quad 1\left(2^{\frac{1}{1}} - 1\right) = 1.0,$$
$$\sum_{i=1}^{2} \frac{C_i}{T_i} = 0.15 \quad \le \quad 2\left(2^{\frac{1}{2}} - 1\right) = 0.828,$$
$$\sum_{i=1}^{3} \frac{C_i}{T_i} = 0.25 \quad \le \quad 3\left(2^{\frac{1}{3}} - 1\right) = 0.780,$$
$$\sum_{i=1}^{4} \frac{C_i}{T_i} = 0.3 \quad \le \quad 4\left(2^{\frac{1}{4}} - 1\right) = 0.757,$$

$$\sum_{i=1}^{5}\frac{C_i}{T_i} = 0.325 \quad \leq \quad 5\left(2^{\frac{1}{5}}-1\right) = 0.743,$$

$$\sum_{i=1}^{6}\frac{C_i}{T_i} = 0.485 \quad \leq \quad 6\left(2^{\frac{1}{6}}-1\right) = 0.735,$$

$$\sum_{i=1}^{7}\frac{C_i}{T_i} = 0.605 \quad \leq \quad 7\left(2^{\frac{1}{7}}-1\right) = 0.729,$$

$$\sum_{i=1}^{8}\frac{C_i}{T_i} = 0.765 \quad \leq \quad 8\left(2^{\frac{1}{8}}-1\right) = 0.724.$$

The basic schedulability test confirms that the seven highest priority tasks are schedulable, but not the eighth task. To determine the schedulability of the eighth task, the scheduling points and loadings for the task are calculated,

$$S_8 = \left\{ kT_j \,\middle|\, j = 1,\ldots,8;\ k = 1,\ldots,\left\lfloor \frac{T_i}{T_j} \right\rfloor \right\} = \{10,20,30,40,50\},$$

$$L_8(10) = \frac{W_8(10)}{10} = \frac{\sum_{j=1}^{8} C_j \left\lceil \frac{10}{T_j} \right\rceil}{10} = \frac{1+2+4+2+1+8+6+8}{10} = \frac{32}{10},$$

$$L_8(20) = \frac{W_8(20)}{20} = \frac{\sum_{j=1}^{8} C_j \left\lceil \frac{20}{T_j} \right\rceil}{20} = \frac{2(1)+2+4+2+1+8+6+8}{20} = \frac{33}{20},$$

$$L_8(30) = \frac{W_8(30)}{30} = \frac{\sum_{j=1}^{8} C_j \left\lceil \frac{30}{T_j} \right\rceil}{30} = \frac{3(1)+2+4+2+1+8+6+8}{30} = \frac{34}{30},$$

$$L_8(40) = \frac{W_8(40)}{40} = \frac{\sum_{j=1}^{8} C_j \left\lceil \frac{40}{T_j} \right\rceil}{40} = \frac{4(1)+2+4+2+1+8+6+8}{40} = \frac{35}{40},$$

$$L_8(50) = \frac{W_8(50)}{50} = \frac{\sum_{j=1}^{8} C_j \left\lceil \frac{50}{T_j} \right\rceil}{50} = \frac{5(1)+2(2)+2(4)+2(2)+2(1)+8+6+8}{50} = \frac{45}{50}.$$

The rate-monotonic workload analysis schedulability test can then be applied,

$$\begin{aligned} L_8 &= \min_{t \in S_8} L_8(t) \leq 1, \\ &= \min\left\{\frac{32}{10}, \frac{33}{20}, \frac{34}{30}, \frac{35}{40}, \frac{45}{50}\right\} \leq 1, \\ &= \frac{35}{40} \leq 1. \end{aligned}$$

Since the workload analysis schedulability test inequality is satisfied, the eighth task of the generic avionics mission system is schedulable. Repeating the workload analysis calculations for the ninth, tenth and eleventh tasks produces the schedulability test results

$$L_9 = \frac{76}{80} \leq 1,$$

$$L_{10} = \frac{100}{100} \leq 1, \text{ and}$$

$$L_{11} = \frac{103}{100} \leq 1.$$

The rate-monotonic workload analysis schedulability test inequality is satisfied for the ninth and tenth generic avionics mission system tasks, and these tasks are schedulable. The eleventh task does not satisfy the schedulability test inequality, and is not schedulable under the rate-monotonic algorithm. It is interesting to note that the sufficient condition established by the basic rate monotonic scheduling test could only guarantee that the seven highest priority tasks could be scheduled, achieving a processor utilisation of 0. 605, when in fact ten tasks could be scheduled, achieving a total processor utilisation of 0. 91. For these ten generic avionics mission system tasks the rate-monotonic algorithm performs well above the theoretical worst-case utilisation limit for ten tasks of 0. 718.

Plotting a rate-monotonic scheduling of the task set illustrates the behaviours of the rate-monotonic algorithm that lead to tasks missing their deadlines. Figure 1 shows such a scheduling for a period of 400 ms, which spans the largest deadline of the task set and is sufficient to illustrate the types of schedule failures that occur. In Figure 1, task rates, and hence priorities, are greatest at the bottom of the figure and decrease monotonically towards the top of the figure, with the *idle* pseudo-task being an indication of when the processor enters an idle state with no task to execute.

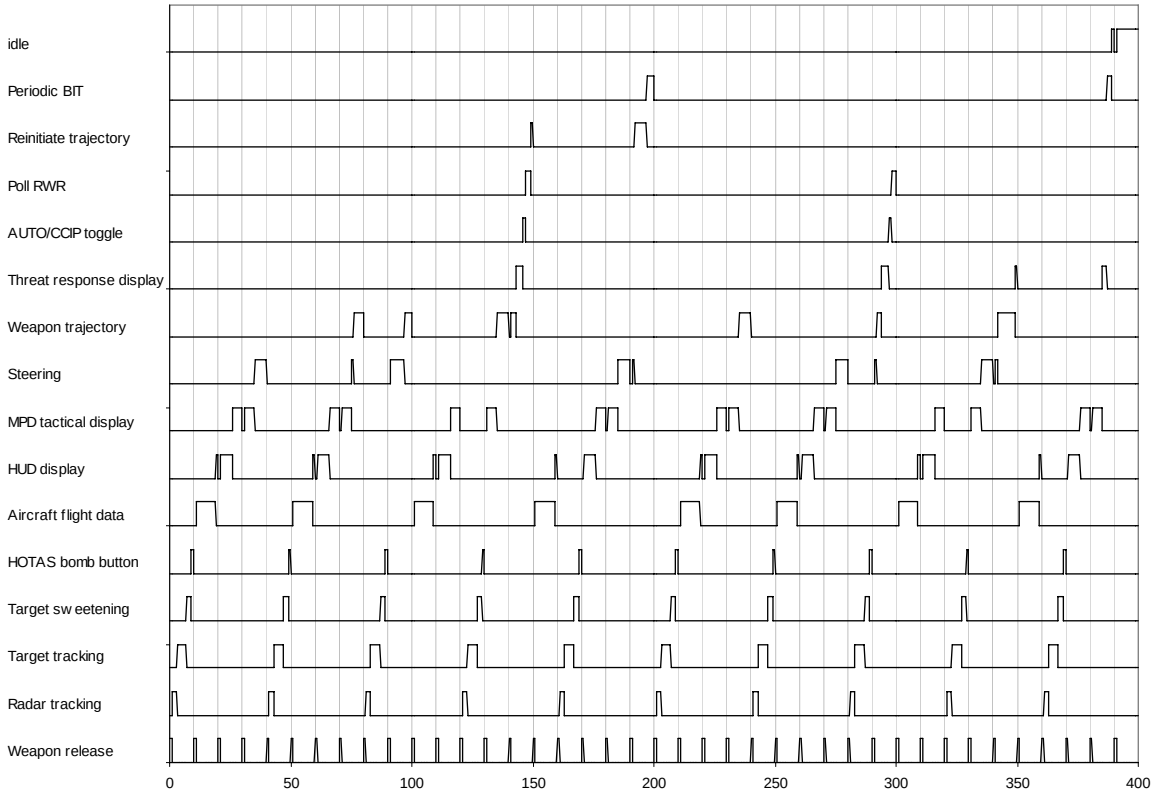

Figure 1. Rate-monotonic scheduling of the generic avionics mission system tasks

The ten highest priority tasks can be seen to be scheduled correctly, meeting their execution deadlines in every period. These tasks have periods from 10 ms up to 100 ms. The *Threat response display* task also has a period of 100 ms, but is not able to run in the first 100 ms and hence misses its first deadline. This illustrates a property of rate-monotonic scheduling, whereby lower priority tasks miss their deadlines under overload conditions, but a schedulable subset of higher priority tasks will always meet their deadlines. This property gives the algorithm a desirable stability under transient overload conditions, making it possible to predict which tasks will not meet their deadlines.

The mechanism by which the algorithm fails is also easily seen in Figure 1. Note how during the first 100 ms period for the *Threat response display* task there are always higher priority tasks ready to run, causing it to miss its deadline. The fixed task priorities allow the 40 ms period tasks to run for a third time, and the 80 ms period task to run for a second time, before the 100 ms period *Threat response display* task is eligible to run. This results in these higher priority tasks completing earlier than necessary, with the consequence that the lower priority *Threat response display* task cannot be executed before its deadline. This characteristic leads to the need to limit processor utilisation to successfully employ rate-monotonic scheduling.

3.1.3 Response Time Analysis

Response time analysis is applicable to any fixed priority scheduling algorithm, and will be used to analyse a deadline-monotonic scheduling of the generic avionics mission system tasks. Deadline-monotonic scheduling is a fixed priority scheduling algorithm that assigns task priorities on a monotonic ordering of task deadlines. When task deadlines are equal to task periods, this priority assignment is equivalent to rate-monotonic prioritisation. The advantage of this algorithm is that it allows task deadlines to be less than or equal to task periods [4], and the generic avionics mission system tasks can be analysed without any variation to the task set specifications.

Response time analysis of deadline-monotonic fixed priority scheduling uses calculations of worst-case response times to determine if tasks are executed before their deadlines. Applying response time calculations as described by Tindell [5], response time R_i for generic avionics mission system task i, comprises a task computation time C_i, and an interference time I_i where task execution is preempted by higher priority tasks,

$$R_i = C_i + I_i .$$

Interference times are bounded by the number of occasions that higher priority tasks can run during the response time of a task. Considering tasks to be ordered by priority, with task *1* having the highest priority, the interference experienced by task i can be stated as

$$I_i = \sum_{j=1}^{i-1} \left\lceil \frac{R_i}{T_j} \right\rceil C_j .$$

This leads to an expression for the response time of task i,

$$R_i = C_i + \sum_{j=1}^{i-1} \left\lceil \frac{R_i}{T_j} \right\rceil C_j ,$$

that can be seen to contain the response time on the left-hand side and right-hand side. Solution is by formation of the response time recurrence relation,

$$R_i^{m+1} = C_i + \sum_{j=1}^{i-1} \left\lceil \frac{R_i^m}{T_j} \right\rceil C_j ,$$

in which an initial value of zero for R_i^m is suitable for the relation to converge to the smallest response time value that satisfies the response time expression. After iterating over the response time recurrence relation until it converges, task schedulability is tested by comparing the worst-case response time to the task deadline,

$R_i \le D_i$.

Applying the response time recurrence relation to the generic avionics mission system, the response time for the highest priority task, task 1, has no interference from higher priority tasks and is given by

$$R_1^1 = C_1 = 1.$$

Schedulability for task 1 is determined by testing that the response time is less than the deadline for the task,

$$R_1 = 1 \le D_1 = 5.$$

Response time for task 2 includes interference due to preemption by task 1,

$$R_2^1 = C_2 + \sum_{j=1}^{1} \left\lceil \frac{R_2^0}{T_j} \right\rceil C_j = 2 + 0(1) = 2,$$

$$R_2^2 = C_2 + \sum_{j=1}^{1} \left\lceil \frac{R_2^1}{T_j} \right\rceil C_j = 2 + 1(1) = 3,$$

$$R_2^3 = C_2 + \sum_{j=1}^{1} \left\lceil \frac{R_2^2}{T_j} \right\rceil C_j = 2 + 1(1) = 3.$$

Comparing task 2 response time to task 2 deadline,

$$R_2 = 3 \le D_2 = 40,$$

task 2 response time is less than its deadline and task 2 is schedulable. Continuing to determine response times for tasks yields the following tests for schedulability,

$$R_3 = 7 \le D_3 = 40, \qquad R_4 = 9 \le D_4 = 40,$$
$$R_5 = 10 \le D_5 = 40, \qquad R_6 = 19 \le D_6 = 50,$$
$$R_7 = 26 \le D_7 = 50, \qquad R_8 = 35 \le D_8 = 50,$$
$$R_9 = 76 \le D_9 = 80, \qquad R_{10} = 100 \le D_{10} = 100,$$
$$R_{11} = 146 \le D_{11} = 100.$$

The response time for the eleventh task is greater than its deadline, and it fails the response time schedulability test. This result is in agreement with the rate-monotonic

workload analysis schedulability test, which also found that the eleventh task was not schedulable. Response time schedulability analysis provides an advantage over workload analysis schedulability testing, in that it is able to quantify the extent by which a task will miss its deadline.

3.2 Non-Preemptive Fixed Priority Scheduling

Non-preemptive fixed priority scheduling avoids the usage of task preemptions by selecting the highest priority task ready for execution at the completion of the current task. This algorithm simplifies scheduler design, but increases response times for high priority tasks, which would otherwise benefit from being able to preempt lower priority tasks. For the task set of the generic avionics mission system, non-preemptive fixed priority scheduling is clearly unable to succeed, since there exist tasks with execution times longer than the deadline for the *Weapon release* task, which can delay this task past its deadline. Analysis by a simple but pessimistic schedulability test is shown in §3.2.1, and by a less pessimistic schedulability test in §3.2.2.

3.2.1 Simple Non-Preemptive Response Time Test

Non-preemptive fixed priority scheduling can be considered to be a special case of preemptive scheduling, where lower priority tasks can block the execution of higher priority tasks until the lower priority task completes its execution [6].

Fidge [6] presents a response time calculation that incorporates non-preemptive blocking of higher priority tasks by lower priority tasks, where response time R_i, for task i, is calculated in terms of computation time C_i, blocking time B_i and interference from higher priority tasks I_i. This can be expressed as

$$R_i = C_i + B_i + I_i,$$

where B_i can be replaced by the maximum computation time of any lower priority task, and I_i can be replaced by a sum of all possible interference during the response time, giving

$$R_i = C_i + \max_{i+1 \le k \le n} C_k + \sum_{j=1}^{i-1} \left\lceil \frac{R_i}{T_j} \right\rceil C_j .$$

The interference term is pessimistic since it assumes that preemption can occur, but is still useful to provide a simple schedulability test. Applying to task 1 of the generic avionics mission system task set,

$$R_1 = C_1 + \max_{2 \le k \le 15} C_k = 1 + 8 = 9,$$

and applying the response time schedulability test

$$R_1 = 9 \leq D_1 = 5,$$

the response time exceeds the deadline and task 1 is not found to be schedulable by the non-preemptive fixed priority algorithm. Testing for schedulability of task 2 requires iteration over the response time recurrence relation until it converges,

$$\begin{aligned}
R_2^0 &= 0, \\
R_2^1 &= C_2 + \max_{3 \leq k \leq 15} C_k = 2 + 8 = 10, \\
R_2^2 &= C_2 + \max_{3 \leq k \leq 15} C_k + \sum_{j=1}^{1} \left\lceil \frac{R_2^1}{T_j} \right\rceil C_j = 2 + 8 + 1 = 11, \\
R_2^3 &= C_2 + \max_{3 \leq k \leq 15} C_k + \sum_{j=1}^{1} \left\lceil \frac{R_2^2}{T_j} \right\rceil C_j = 2 + 8 + 2(1) = 12, \\
R_2^4 &= C_2 + \max_{3 \leq k \leq 15} C_k + \sum_{j=1}^{1} \left\lceil \frac{R_2^3}{T_j} \right\rceil C_j = 2 + 8 + 2(1) = 12.
\end{aligned}$$

Applying the response time schedulability test for task 2,

$$R_2 = 12 \leq D_2 = 40,$$

the response time does not exceed the deadline and the task is schedulable by the non-preemptive fixed priority algorithm. Similarly determining response times and applying the response time schedulability test for the remaining generic avionics mission system tasks,

$$\begin{aligned}
R_3 &= 16 \leq D_3 = 40, & R_4 &= 18 \leq D_4 = 40, \\
R_5 &= 19 \leq D_5 = 40, & R_6 &= 28 \leq D_6 = 50, \\
R_7 &= 35 \leq D_7 = 50, & R_8 &= 68 \leq D_8 = 50, \\
R_9 &= 94 \leq D_9 = 80, & R_{10} &= 142 \leq D_{10} = 100, \\
R_{11} &= 194 \leq D_{11} = 100, & R_{12} &= 198 \leq D_{12} = 200, \\
R_{13} &= 200 \leq D_{13} = 200, & R_{14} &= 393 \leq D_{14} = 400, \\
R_{15} &= 393 \leq D_{15} = 400.
\end{aligned}$$

Due to the pessimistic interference value used in the construction of this test, these results indicate that only tasks 2, 3, 4, 5, 6, 7, 12, 13, 14 and 15 can be successfully scheduled by the non-preemptive fixed priority algorithm. Task 8 is also schedulable, as can be determined by the following test.

3.2.2 Non-Preemptive Response Time Test

A less pessimistic schedulability test can be constructed by disallowing preemption of interfering tasks, as described by Fidge [6]. Consider task response time R_i to consist of a computation time C_i and a release time r_i,

$$R_i = C_i + r_i .$$

Release time will comprise a worst-case blocking time from a lower priority task and interference from non-preemptive arrivals of higher priority tasks,

$$r_i = B_i + I_i ,$$
$$r_i = \max_{i+1 \le k \le n} C_k + \sum_{j=1}^{i-1} \left(\left\lfloor \frac{r_i}{T_j} \right\rfloor + 1 \right) C_j .$$

Solution of release time is by formation of a recurrence relation,

$$r_i^{m+1} = \max_{i+1 \le k \le n} C_k + \sum_{j=1}^{i-1} \left(\left\lfloor \frac{r_i^m}{T_j} \right\rfloor + 1 \right) C_j .$$

Release time for task 1 of the generic avionics mission system task set is found through solution of the recurrence relation,

$$r_1^0 = 0,$$
$$r_1^1 = \max_{2 \le k \le 15} C_k = 8.$$

Response time for task 1 can then be determined and the response time schedulability test applied,

$$R_1 = C_1 + r_1 = 1 + 8 = 9,$$
$$R_1 = 9 \le D_1 = 5.$$

The response time for task 1 fails to satisfy the response time test inequality, so task 1 is not found to meet its deadline under the non-preemptive fixed priority algorithm.

Performing the calculations for task 2,

$$r_2^0 = 0,$$

$$r_2^1 = \max_{3\le k\le 15} C_k + \sum_{j=1}^{1}\left(\left\lfloor \frac{r_2^0}{T_j}\right\rfloor + 1\right)C_j = 8+1 = 9,$$

$$r_2^2 = \max_{3\le k\le 15} C_k + \sum_{j=1}^{1}\left(\left\lfloor \frac{r_2^1}{T_j}\right\rfloor + 1\right)C_j = 8+1 = 9,$$

$$R_2 = C_2 + r_2 = 2+9 = 11,$$

$$R_2 = 11 \le D_2 = 40,$$

task 2 satisfies the response time schedulability test, and will always meet its deadline.

Repeating the calculations for task 3,

$$r_3^0 = 0,$$

$$r_3^1 = \max_{4\le k\le 15} C_k + \sum_{j=1}^{2}\left(\left\lfloor \frac{r_3^0}{T_j}\right\rfloor + 1\right)C_j = 8+1+2 = 11,$$

$$r_3^2 = \max_{4\le k\le 15} C_k + \sum_{j=1}^{2}\left(\left\lfloor \frac{r_3^1}{T_j}\right\rfloor + 1\right)C_j = 8+2(1)+2 = 12,$$

$$r_3^3 = \max_{4\le k\le 15} C_k + \sum_{j=1}^{2}\left(\left\lfloor \frac{r_3^2}{T_j}\right\rfloor + 1\right)C_j = 8+2(1)+2 = 12,$$

$$R_3 = C_3 + r_3 = 4+12 = 16,$$

$$R_3 = 16 \le D_3 = 40,$$

task 3 also satisfies the response time schedulability test. Response time tests for the remaining generic avionics mission system tasks have been determined as follows,

$$R_4 = 18 \le D_4 = 40, \qquad R_5 = 19 \le D_5 = 40,$$

$$R_6 = 27 \le D_6 = 50, \qquad R_7 = 34 \le D_7 = 50,$$

$$R_8 = 42 \le D_8 = 50, \qquad R_9 = 83 \le D_9 = 80,$$

$$R_{10} = 106 \le D_{10} = 100, \qquad R_{11} = 152 \le D_{11} = 100,$$

$$R_{12} = 198 \le D_{12} = 200, \qquad R_{13} = 200 \le D_{13} = 200,$$

$$R_{14} = 205 \le D_{14} = 400, \qquad R_{15} = 390 \le D_{15} = 400.$$

This response time schedulability test indicates that tasks 2, 3, 4, 5, 6, 7, 8, 12, 13, 14 and 15 will always meet their deadlines under the non-preemptive fixed priority algorithm. Figure 2 shows the higher levels of jitter that this algorithm introduces for high priority tasks, causing the *Weapon release* task to miss its deadline, while similarity to the

rate-monotonic and deadline-monotonic algorithms is evident in the manner in which the unschedulable tasks miss their deadlines.

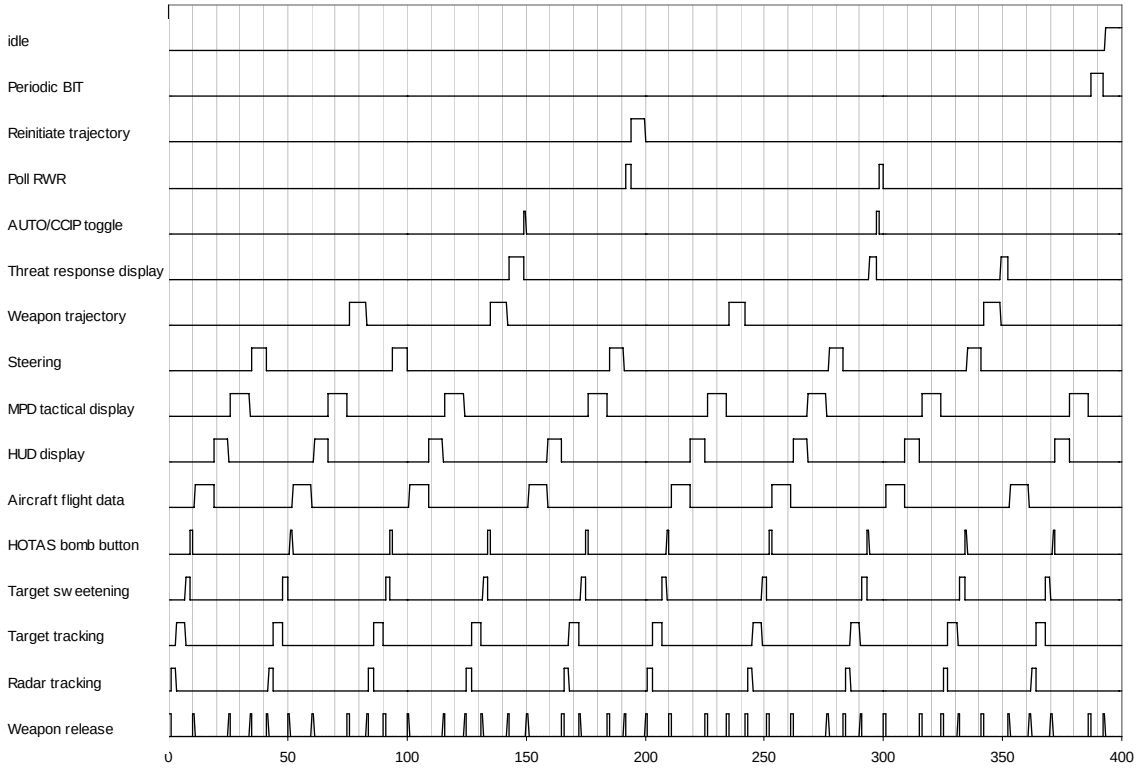

Figure 2. Non-preemptive fixed priority scheduling of generic avionics mission system tasks

3.3 Asynchronous Fixed Priority Scheduling

Synchronous fixed priority scheduling of independent tasks assumes simultaneous arrivals of all tasks at time zero, providing a critical instant [2] from where all tasks will experience their worst case response times. Analysis of synchronous systems by Liu and Layland [2] found that processor utilisation needed to be bounded to ensure that task deadlines would be met for task arrivals at critical instants.

Asynchronous fixed priority scheduling does not require tasks with independent execution timing to have simultaneous arrivals at time zero, allowing the first arrival of a task to be delayed by an arbitrary phase offset. Tasks that have a harmonic relationship in their invocation periods will maintain a predictable phase relationship between their arrival times, which can reduce the number of tasks having simultaneous arrivals, and allow a higher level of processor utilisation.

A sufficient and necessary schedulability analysis for asynchronous systems has been reported by Tindell [7], using a concept of transactions containing tasks with related

timings, but is computationally infeasible for non-trivial problems [7]. Leung and Whitehead [8] have studied the complexity of asynchronous scheduling, and prove that if the schedulability of an identified partial schedule is determined, then the schedulability of the system for all time is proved. The length of the partial schedule is given by twice the lowest common multiple of all task periods, which has a worst case exponential complexity, and may result in analysis not being computationally feasible for systems with large numbers of tasks with relatively prime periods.

A phase offset specification can be added to tasks of the generic avionics mission system as shown in Table 2. Tasks retain a deadline-monotonic task prioritisation, while selection of phase offsets has been made to evenly distribute the arrival times of tasks of equal periodicity.

Table 2. Generic avionics mission system tasks with phase offsets

Task Name	**Computation Time (ms)**	**Phase Offset (ms)**	**Period (ms)**	**Deadline (ms)**
Weapon release	1	0	10	5
Radar tracking	2	0	40	
Target tracking	4	10	40	
Target sweetening	2	20		40
HOTAS bomb button	1	30		40
Aircraft flight data	8	0	50 (55)	
HUD display	6	16	50 (52)	
MPD tactical display	8	32	50 (52)	
Steering	6	20	80	
Weapon trajectory	7	0	100	
Threat response display	3	50		100
AUTO/CCIP toggle	1	0		200
Poll RWR	2	100	200	
Reinitiate trajectory	6	0		400
Periodic BIT	5	0	1000	400

Analysis of asynchronous task set schedulability is by application of the fixed priority scheduling algorithm for a time period spanning the Leung and Whitehead partial schedule [8], which starts at the largest phase offset value. Schedule start-up at time zero represents atypical behaviour since phase-shifted tasks are only scheduled for their first execution after their phase offset delay, and has been included in the analysis even though it is outside of the required partial schedule. Note that if all task periods were relatively prime, schedulability would revert to the case of synchronous tasks, since synchronous arrival of all tasks would occur somewhere within the partial schedule.

Schedulability analysis of the asynchronous task set involves a lengthy sequence of calculations, which will be developed into an iterative algorithm. To assist with algorithmic expression of the schedulability analysis, vectors and matrices are used for a number of calculation variables. The variable $n = 15$ is used to refer to the number of tasks, and the variable m is used as a local variable indicating the number of rows in a

matrix or column vector. Generic avionics mission system task periods, computation times, and phase offsets are assigned to vectors

$$T = \begin{bmatrix} 10 \\ 40 \\ 40 \\ 40 \\ 40 \\ 50 \\ 50 \\ 50 \\ 80 \\ 100 \\ 100 \\ 200 \\ 200 \\ 400 \\ 1000 \end{bmatrix}, \quad C = \begin{bmatrix} 1 \\ 2 \\ 4 \\ 2 \\ 1 \\ 8 \\ 6 \\ 8 \\ 6 \\ 7 \\ 3 \\ 1 \\ 2 \\ 6 \\ 5 \end{bmatrix}, \quad \text{and} \quad P = \begin{bmatrix} 0 \\ 0 \\ 10 \\ 20 \\ 30 \\ 0 \\ 16 \\ 32 \\ 20 \\ 0 \\ 50 \\ 0 \\ 100 \\ 0 \\ 0 \end{bmatrix}.$$

The test period to be considered in the analysis will be the Leung and Whitehead partial schedule [8], expanded to begin at time zero. The end point of the partial schedule is the largest phase offset delay plus twice the lowest common multiple of all task periods, giving a test period T_t of

$$T_t = \max_{1 \le i \le n} P_i + 2 * \text{lcm}(T_1, \ldots, T_n) = 100 + 2 * 2000 = 4100 \,\text{ms}.$$

Task arrival times are calculated for the complete test period, giving a set of arrival times for each generic avionics mission system task, using

$$A_i = \left\{ kT_i + P_i \,\middle|\, i = 1, \ldots, n; k = 0, \ldots, \left\lceil \frac{T_t - P_i}{T_i} \right\rceil - 1 \right\},$$

which produces the sets

$$A_1 = \{0,10,\ldots,4090\}, \qquad A_2 = \{0,40,\ldots,4080\},$$
$$A_3 = \{10,50,\ldots,4090\}, \qquad A_4 = \{20,60,\ldots,4060\},$$
$$A_5 = \{30,70,\ldots,4070\}, \qquad A_6 = \{0,50,\ldots,4050\},$$

$A_7 = \{16,66,\ldots,4066\}$, $A_8 = \{32,82,\ldots,4082\}$,
$A_9 = \{20,100,\ldots,4020\}$, $A_{10} = \{0,100,\ldots,4000\}$,
$A_{11} = \{50,150,\ldots,4050\}$, $A_{12} = \{0,200,\ldots,4000\}$,
$A_{13} = \{100,300,\ldots,3900\}$, $A_{14} = \{0,400,\ldots,4000\}$,
$A_{15} = \{0,1000,2000,3000,4000\}$.

The union of individual task arrival time sets creates a set A, containing all task arrival times in the test period,

$$A = \bigcup_{i=1}^{n} A_i = \{0,10,\ldots,4090\} \cup \{16,66,\ldots,4066\} \cup \{32,82,\ldots,4082\}.$$

Task response times are calculated by first allocating arrived tasks to a service queue, from where tasks receive service in priority order. Task servicing may be preempted by arrival and servicing of higher priority tasks. Task response times are calculated as the elapsed time between task arrival and the completion of task service. Response times will vary according to the relative phasings of tasks, and the maximum task response time occurring in the test period provides the worst-case response time measurement used to determine schedulability for a task.

An array S is used to represent the task service queue. Each row of this array holds five parameters for a task awaiting service: task priority; task arrival time; task completion time; task response time and task computation time. Task service queue S is maintained in priority order by inserting newly arrived tasks after queued tasks of equal or higher priority. Row 1 of S represents the head of the service queue and contains parameters for the task to be serviced. On completion of task service, task parameter information is deleted from the head of service queue S, and is appended to the end of array F, which is used to store parameters of tasks that have completed service. Array F will contain parameters for all completed tasks, ordered by task completion times.

A schedule time is maintained in variable t, and the time step that has occurred from the preceding calculation is maintained in variable dt. Initially, $t = 0$ and $dt = 0$ at the commencement of analysis. The task service queue and completed task array are initially empty, $S = [\ ]$ and $F = [\ ]$. Iterative calculations are needed to analyse scheduling behaviour for the entire test period, and are introduced through analysis of schedule behaviour for the first 10 ms of schedule execution. An iterative algorithm is then presented that can perform the analysis for the complete test period. To support algorithm development, an assignment operator $:=$ is used to indicate that a variable is assigned a new value, which may recursively depend on the current value of the variable.

At commencement of schedule analysis, $S = [\ ]$ and there is no queued task awaiting service. Schedule time $t = 0$, and existence of an element $0 \in A$ indicates that task arrivals

occur. Element 0 is removed from A, leaving A to contain only task arrival times not yet reached,

$$A := A - \{0\} := \{10,20,\ldots,4090\} \cup \{16,66,\ldots,4066\} \cup \{32,82,\ldots,4082\}.$$

Tasks *Weapon release*, *Radar tracking*, *Aircraft flight data*, *Weapon trajectory*, *AUTO/CCIP toggle*, *Reinitiate trajectory* and *Periodic BIT* arrive at time $t = 0$. Arriving tasks need to be queued in task service queue S according to task priorities, and this operation is assisted by defining a function

$$\text{enqueue}(S, A_i, p, a, c, r, d) = \begin{cases} S & , a \notin A_i \\ \begin{bmatrix} p & a & c & r & d \end{bmatrix} & , S = [\ ] \\ \begin{bmatrix} p & a & c & r & d \\ S_{(1)(1)} & S_{(1)(2)} & S_{(1)(3)} & S_{(1)(4)} & S_{(1)(5)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ S_{(m)(1)} & S_{(m)(2)} & S_{(m)(3)} & S_{(m)(4)} & S_{(m)(5)} \end{bmatrix} & , p < S_{(1)(1)} \\ \begin{bmatrix} S_{(1)(1)} & S_{(1)(2)} & S_{(1)(3)} & S_{(1)(4)} & S_{(1)(5)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ S_{(m)(1)} & S_{(m)(2)} & S_{(m)(3)} & S_{(m)(4)} & S_{(m)(5)} \\ p & a & c & r & d \end{bmatrix} & , p \geq S_{(m)(1)}. \\ \begin{bmatrix} S_{(1)(1)} & S_{(1)(2)} & S_{(1)(3)} & S_{(1)(4)} & S_{(1)(5)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ S_{(k)(1)} & S_{(k)(2)} & S_{(k)(3)} & S_{(k)(4)} & S_{(k)(5)} \\ p & a & c & r & d \\ S_{(k+1)(1)} & S_{(k+1)(2)} & S_{(k+1)(3)} & S_{(k+1)(4)} & S_{(k+1)(5)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ S_{(m)(1)} & S_{(m)(2)} & S_{(m)(3)} & S_{(m)(4)} & S_{(m)(5)} \end{bmatrix} & , S_{(k)(1)} \leq p < S_{(k+1)(1)} \end{cases}$$

Applying function enqueue to assign arriving task information to the task service queue S,

$$S := \{\text{enqueue}(S, A_i, i, t, 0, 0, C_i) \quad , i = 1, \ldots, n$$
$$:= \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 2 & 0 & 0 & 0 & 2 \\ 6 & 0 & 0 & 0 & 8 \\ 10 & 0 & 0 & 0 & 7 \\ 12 & 0 & 0 & 0 & 1 \\ 14 & 0 & 0 & 0 & 6 \\ 15 & 0 & 0 & 0 & 5 \end{bmatrix}.$$

Function enqueue inserts task parameters into the task service queue S, for each of the seven arriving tasks. Tasks are inserted in order of priorities as indicated by column 1 of S. Column 2 contains task arrival times, which are all zero for the seven arrived tasks. Columns 3 and 4 hold task completion times and response times that are not yet known, and are zeroed until determined. Column 5 holds the remaining computation times needed to complete the tasks, initially set equal to task computation times for arriving tasks.

Schedule time can be advanced to the next task arrival time or completion of task service, whichever occurs first. The time step for this advancement is given by

$$\begin{aligned} dt &= \begin{cases} \min(A) - t & , S = [\ ] \\ S_{(1)(5)} & , t + S_{(1)(5)} < \min(A) \\ \min(A) - t & , t + S_{(1)(5)} \geq \min(A) \end{cases} \\ &= S_{(1)(5)} \\ &= 1, \end{aligned}$$

and schedule time is advanced to

$$t := t + dt := 0 + 1 := 1.$$

The priority 1 *Weapon release* task at the head of task service queue S completes at $t = 1$, and requires its parameter data fields to be updated. This task may then be removed from the service queue and placed into the empty completed task array F. Updating task completion time, response time and computation time fields,

$$S_{(1)(3)} = t = 1,$$
$$S_{(1)(4)} = S_{(1)(3)} - S_{(1)(2)} = 1,$$
$$S_{(1)(5)} = 0.$$

Copying task parameter data into the completed task array F, and removing it from the task service queue,

$$F \coloneqq \begin{bmatrix} S_{(1)(1)} & S_{(1)(2)} & S_{(1)(3)} & S_{(1)(4)} & S_{(1)(5)} \end{bmatrix} \coloneqq \begin{bmatrix} 1 & 0 & 1 & 1 & 0 \end{bmatrix},$$

$$S \coloneqq \begin{bmatrix} S_{(2)(1)} & S_{(2)(2)} & S_{(2)(3)} & S_{(2)(4)} & S_{(2)(5)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ S_{(7)(1)} & S_{(7)(2)} & S_{(7)(3)} & S_{(7)(4)} & S_{(7)(5)} \end{bmatrix} \coloneqq \begin{bmatrix} 2 & 0 & 0 & 0 & 2 \\ 6 & 0 & 0 & 0 & 8 \\ 10 & 0 & 0 & 0 & 7 \\ 12 & 0 & 0 & 0 & 1 \\ 14 & 0 & 0 & 0 & 6 \\ 15 & 0 & 0 & 0 & 5 \end{bmatrix}.$$

Six tasks remain in the service queue, and schedule time can again be advanced to the next task arrival time or completion of task service. The time step for this advancement is given by

$$\begin{aligned} dt &= \begin{cases} \min(A) - t & , S = [\ ] \\ S_{(1)(5)} & , t + S_{(1)(5)} < \min(A) \\ \min(A) - t & , t + S_{(1)(5)} \geq \min(A) \end{cases} \\ &= S_{(1)(5)} \\ &= 2, \end{aligned}$$

and schedule time is advanced to

$$t \coloneqq t + dt \coloneqq 1 + 2 \coloneqq 3.$$

The priority 2 *Radar tracking* task completes at $t = 3$, and requires its parameter data fields to be updated in task service queue S. This task may then be removed from the service queue and be appended to the completed task array F. Updating task completion time, response time and computation time fields,

$$\begin{aligned} S_{(1)(3)} &= t = 3, \\ S_{(1)(4)} &= S_{(1)(3)} - S_{(1)(2)} = 3 - 0 = 3, \\ S_{(1)(5)} &= 0. \end{aligned}$$

Copying task parameter data into the completed task array F and removing it from the task service queue,

$$F \coloneqq \begin{bmatrix} F_{(1)(1)} & F_{(1)(2)} & F_{(1)(3)} & F_{(1)(4)} & F_{(1)(5)} \\ S_{(1)(1)} & S_{(1)(2)} & S_{(1)(3)} & S_{(1)(4)} & S_{(1)(5)} \end{bmatrix} \coloneqq \begin{bmatrix} 1 & 0 & 1 & 1 & 0 \\ 2 & 0 & 3 & 3 & 0 \end{bmatrix},$$

$$S := \begin{bmatrix} S_{(2)(1)} & S_{(2)(2)} & S_{(2)(3)} & S_{(2)(4)} & S_{(2)(5)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ S_{(6)(1)} & S_{(6)(2)} & S_{(6)(3)} & S_{(6)(4)} & S_{(6)(5)} \end{bmatrix} := \begin{bmatrix} 6 & 0 & 0 & 0 & 8 \\ 10 & 0 & 0 & 0 & 7 \\ 12 & 0 & 0 & 0 & 1 \\ 14 & 0 & 0 & 0 & 6 \\ 15 & 0 & 0 & 0 & 5 \end{bmatrix}.$$

Five tasks remain in the service queue, and schedule time is again advanced to the next task arrival time or completion of task service. The time step for this advancement is given by

$$\begin{aligned} dt &= \begin{cases} \min(A) - t & , S = [\ ] \\ S_{(1)(5)} & , t + S_{(1)(5)} < \min(A) \\ \min(A) - t & , t + S_{(1)(5)} \geq \min(A) \end{cases} \\ &= \min(A) - t \\ &= 10 - 3 \\ &= 7, \end{aligned}$$

and schedule time is advanced to

$$t := t + dt := 3 + 7 := 10.$$

The priority 1 *Weapon release* task and priority 3 *Target tracking* task arrive at $t = 10$, and will both preempt the incomplete priority 6 *Aircraft flight data* task. Element 10 is removed from the set of all arrival times A, leaving A to contain only task arrival times not yet reached,

$$A := A - \{10\} := \{20,30,\ldots,4090\} \cup \{16,66,\ldots,4066\} \cup \{32,82,\ldots,4082\}.$$

The incomplete *Aircraft flight data* task must have its outstanding computation time requirements updated in the task service queue S,

$$S_{(1)(5)} := S_{(1)(5)} - dt := 8 - 7 := 1.$$

Arriving tasks *Weapon release* and *Target tracking* can then be inserted into the task service queue S,

$$S := \{\text{enqueue}(S, A_i, i, t, 0, 0, C_i) \quad , i = 1, \ldots, n$$

$$:= \begin{bmatrix} 1 & 10 & 0 & 0 & 1 \\ 3 & 10 & 0 & 0 & 4 \\ 6 & 0 & 0 & 0 & 1 \\ 10 & 0 & 0 & 0 & 7 \\ 12 & 0 & 0 & 0 & 1 \\ 14 & 0 & 0 & 0 & 6 \\ 15 & 0 & 0 & 0 & 5 \end{bmatrix}.$$

Schedule time can again be advanced, task parameter data fields be updated, completed tasks be moved from the service queue to the completed tasks array, and arriving tasks be enqueued for service in priority order, until all task arrivals for the test period have been serviced. On completion of servicing of all task arrivals, the completed task array F will contain response time data for every task execution. Worst-case response times for each generic avionics mission system task are found by identifying the maximum task response times for the test period,

$$R_i := \max\left\{F_{(j)(4)} \middle| j = 1, \ldots, m; F_{(j)(1)} = i\right\}.$$

An algorithmic expression describing the complete schedulability test calculation is presented in Figure 3, in terms of the variables, functions and data structures already introduced. This algorithm has been used to complete the evaluation of worst-case response times. Applying the response time schedulability test for each task,

$R_1 = 1 \le D_1 = 5$, $R_2 = 3 \le D_2 = 40$, $R_3 = 5 \le D_3 = 40$,
$R_4 = 3 \le D_4 = 40$, $R_5 = 2 \le D_5 = 40$, $R_6 = 16 \le D_6 = 50$,
$R_7 = 11 \le D_7 = 50$, $R_8 = 14 \le D_8 = 50$, $R_9 = 28 \le D_9 = 80$,
$R_{10} = 75 \le D_{10} = 100$, $R_{11} = 49 \le D_{11} = 100$, $R_{12} = 79 \le D_{12} = 200$,
$R_{13} = 80 \le D_{13} = 200$, $R_{14} = 200 \le D_{14} = 400$, $R_{15} = 300 \le D_{15} = 400$.

$t := 0$

$dt := 0$

while $A \neq \{\ \}$ or $S \neq [\ ]$

$$dt := \begin{cases} \min(A) - t & , S = [\ ] \\ S_{(1)(5)} & , t + S_{(1)(5)} < \min(A) \\ \min(A) - t & , t + S_{(1)(5)} \geq \min(A) \end{cases}$$

$t := t + dt$

$$A := \begin{cases} A & , t \notin A \\ A - \{t\} & , t \in A \end{cases}$$

if $S \neq [\ ]$

$S_{(1)(5)} := S_{(1)(5)} - dt$

if $S_{(1)(5)} = 0$

$S_{(1)(3)} := t$

$S_{(1)(4)} := S_{(1)(3)} - S_{(1)(2)}$

$S_{(1)(5)} := 0$

$$F := \begin{bmatrix} F_{(1)(1)} & F_{(1)(2)} & F_{(1)(3)} & F_{(1)(4)} & F_{(1)(5)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ F_{(m)(1)} & F_{(m)(2)} & F_{(m)(3)} & F_{(m)(4)} & F_{(m)(5)} \\ S_{(1)(1)} & S_{(1)(2)} & S_{(1)(3)} & S_{(1)(4)} & S_{(1)(5)} \end{bmatrix}$$

$$S := \begin{bmatrix} S_{(2)(1)} & S_{(2)(2)} & S_{(2)(3)} & S_{(2)(4)} & S_{(2)(5)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ S_{(m)(1)} & S_{(m)(2)} & S_{(m)(3)} & S_{(m)(4)} & S_{(m)(5)} \end{bmatrix}$$

end if

end if

$S := \{\text{enqueue}(S, A_i, i, t, 0, 0, C_i); i = 1, \ldots, n$

end while

$$R_i := \left\{\max\left\{F_{(j)(4)} \middle| j = 1, \ldots, m; F_{(j)(1)} = i\right\}\right. \quad , i = 1, \ldots, n$$

Figure 3 Asynchronous fixed priority schedulability test algorithm

The schedulability test reveals that no tasks exceed their deadlines for the asynchronous fixed priority scheduling of generic avionics mission system tasks, producing a fixed priority scheduling solution that was unachievable by synchronous fixed priority scheduling. The superior performance of asynchronous scheduling over synchronous scheduling results when the selected task phasings avoid simultaneous arrivals of all tasks and more evenly spread task arrivals to eliminate loading peaks that restrict processor

utilisation. Figure 4 shows an asynchronous fixed priority scheduling of the generic avionics mission system tasks for a period of 400 ms.

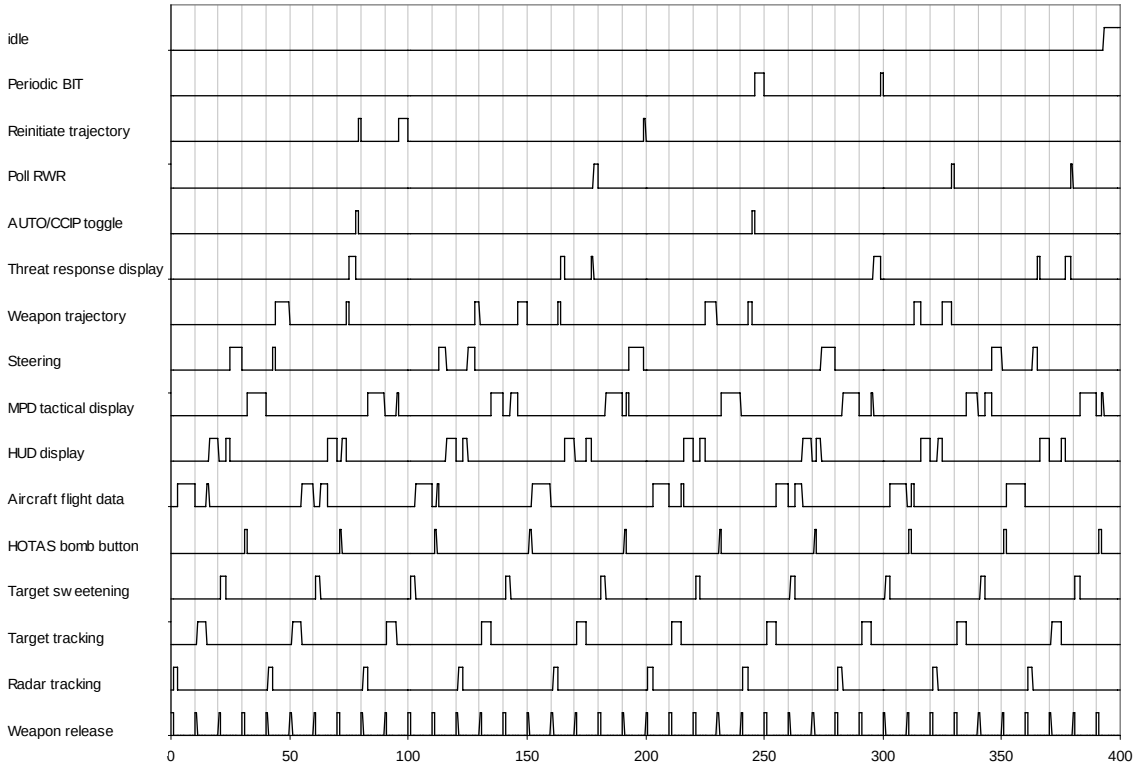

Figure 4. Asynchronous fixed priority scheduling of the generic avionics mission system tasks

3.4 Dynamic Priority Scheduling

The earliest-deadline-first dynamic priority scheduling algorithm provides an optimal scheduling solution where the use of dynamic task priorities is acceptable [2]. This algorithm preemptively selects the task to be run on the criteria of the task having the smallest time remaining until its execution deadline, and is able to fully utilise processor capacity. Implementation of this algorithm involves higher scheduler overheads than fixed priority scheduling algorithms, due to the need to dynamically calculate task priorities before queueing tasks for execution.

To satisfy the assumptions needed to apply the schedulability test derived by Liu and Layland [2], it will be assumed that the deadline for the *Weapon release* task is 10 ms, and that the deadline for the *Periodic BIT* task is 1000 ms. Schedulability analysis is straightforward, requiring only a test of processor utilisation, since the algorithm is only limited by processor capacity. Calculating a processor utilisation test for the fifteen generic avionics mission system tasks,

$$\sum_{i=1}^{15} \frac{C_i}{T_i} = \frac{1}{10} + \frac{2+4+2+1}{40} + \frac{8+6+8}{50} + \frac{6}{80} + \frac{7+3}{100} + \frac{1+2}{200} + \frac{6}{400} + \frac{5}{1000} = 0.975 \leq 1,$$

the total utilisation of 0.975 does not exceed processor capacity and the tasks are schedulable by the earliest-deadline-first algorithm. Figure 5 shows a plot of an earliest-deadline-first scheduling of the generic avionics mission system tasks. It can be seen that this algorithm preserves the low jitter for high priority tasks that the rate-monotonic algorithm produces, but does not run tasks earlier than needed if this would result in another task missing its deadline. Idle processor time observable at the end of the 400 ms period is able to be used without restriction, anywhere in the period, offering maximum flexibility for any additional processing added to the task set. If the algorithm was subjected to a transient overload it is not possible to predict which tasks would miss their deadlines.

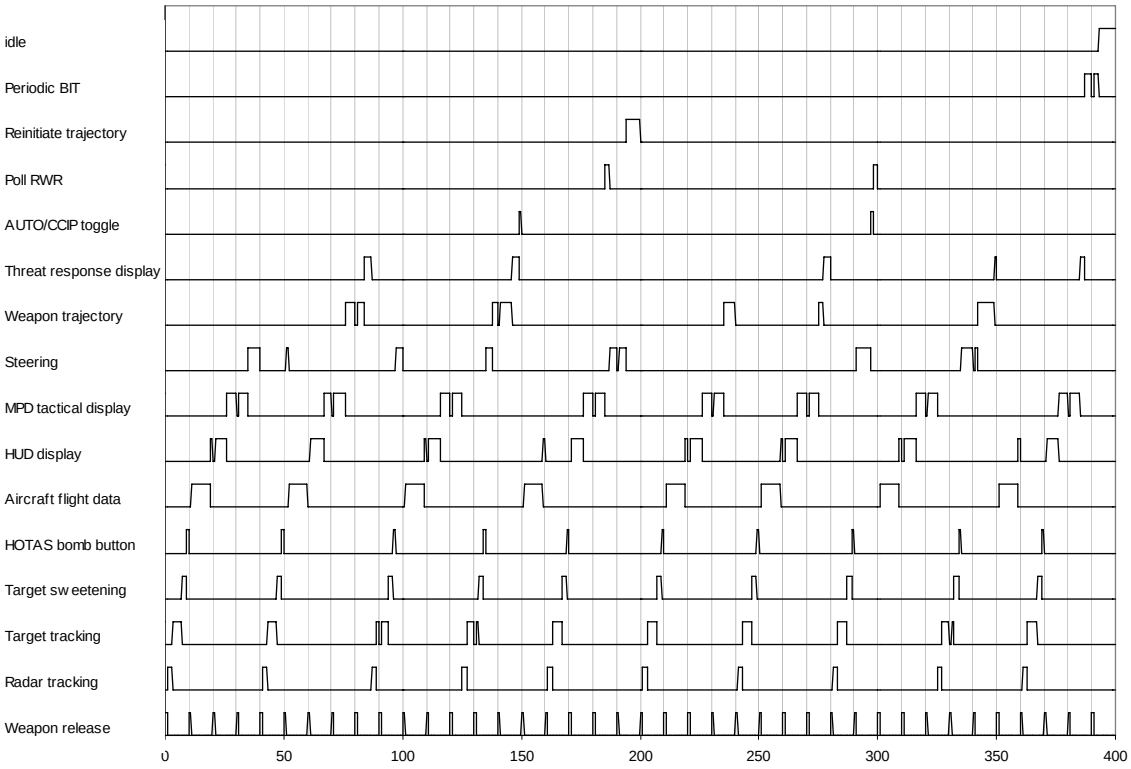

Figure 5. Earliest-deadline-first scheduling of the generic avionics mission system tasks

3.5 Static Scheduling

Statically determined schedules are used where deterministic operation is required, or where the run-time overheads of dynamic scheduling can not be accommodated. Use of a non-preemptive statically determined schedule eliminates much of the need for a run-time operating system, offering the highest possible levels of throughput. Development of the

execution schedule becomes a design activity highly dependent on the execution-time performance of the tasks being scheduled. This leads to schedule design, implementation and testing activity being required to accommodate any change to any task in the task set.

To make a schedule design manageable it is desirable to achieve a reasonably small lowest common multiple of all task periods, since beyond this time the schedule is simply repeated. Best case for design ease occurs when all tasks are scheduled at harmonic frequencies, and the period of the schedule design equals the period of the lowest frequency task. For the generic avionics mission system, three tasks have had their periods shortened to multiples of the shortest task period, to reduce the period of the schedule design. The resulting lowest common multiple for all task periods was 2000 ms. Schedulability analysis involves examining the response time for every task invocation in the schedule, to identify the worst-case response time for each task. Table 3 shows worst-case response times for a static scheduling of the generic avionics mission system tasks, and all task response times can be seen to not exceed task deadlines.

Table 3. Static schedule worst-case response times

Task Name	**Computation Time (ms)**	**Period (ms)**	**Deadline (ms)**	**Response Time (ms)**
Weapon release	1	10	5	4
Radar tracking	2	40		5
Target tracking	4	40		25
Target sweetening	2		40	27
HOTAS bomb button	1		40	35
Aircraft flight data	8	50 (55)		13
HUD display	6	50 (52)		22
MPD tactical display	8	50 (52)		33
Steering	6	80		62
Weapon trajectory	7	100		93
Threat response display	3		100	99
AUTO/CCIP toggle	1		200	120
Poll RWR	2	200		189
Reinitiate trajectory	6		400	50
Periodic BIT	5	1000	400	250

Figure 6 illustrates the first 400 ms of the static schedule, in which jitter for the high rate *Weapon release* task is more pronounced due to the unavailability of preemption. The total idle processor time remains the same as for other successful scheduling algorithms, but to use it at other locations within the schedule requires redesign of the schedule and is not guaranteed to be possible.

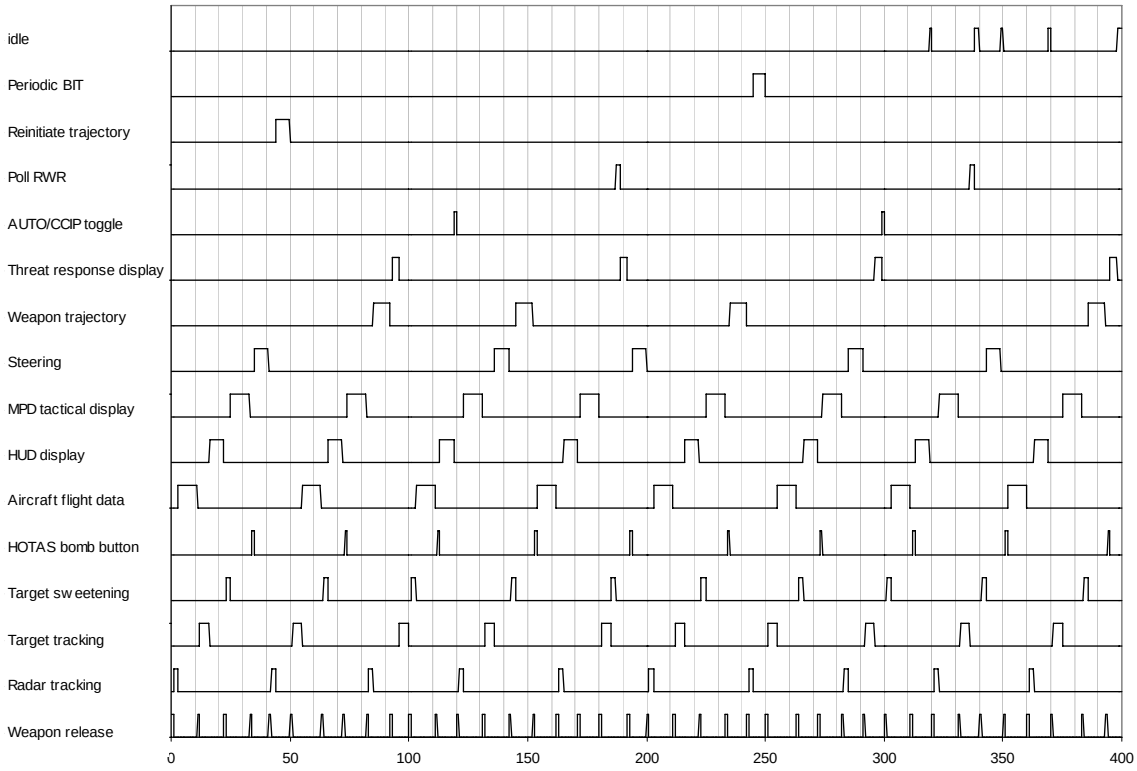

Figure 6. A static scheduling of the generic avionics mission system tasks

4. Conclusions

Established approaches to real-time task scheduling have been applied to a set of generic avionics mission system tasks representative of a heavily loaded system. The scheduling algorithms have been analysed using a range of schedulability analyses, demonstrating the application of the analyses and determining the performance of the scheduling algorithms.

The widely used fixed priority preemptive scheduling algorithm has been applied, and three analysis methods have been demonstrated. The basic rate-monotonic schedulability test was unable to confirm schedulability for the task set, this test is only able to confirm schedulability for less heavily loaded systems. The workload analysis schedulability test is always able to determine schedulability for this scheduling algorithm, and found that the task set cannot be successfully scheduled. Similarly, the response time analysis schedulability test is always able to determine schedulability, and also indicates that the task set is not schedulable. The fixed priority algorithm does demonstrate predictable performance when overloaded, ensuring high priority tasks are completed in preference to low priority tasks, but is not suitable for the heavily loaded system.

Fixed priority non-preemptive scheduling has been subjected to two separate analyses based on extensions of response time analysis. A simple non-preemptive response time test found that five tasks would miss their deadlines, but the results of the test are pessimistic. A less pessimistic non-preemptive response time test found that only four tasks would miss their deadlines. The inability to preempt a low priority task in order to execute a high priority task is a major weakness in fixed priority non-preemptive scheduling, leading to increased jitter and response times for high priority tasks, and makes this algorithm unsuitable for the heavily loaded system.

Asynchronous fixed priority scheduling has been subjected to an analysis that determines worst-case response times to establish schedulability. Although the schedulability test has a worst-case exponential computational complexity, in practical applications the complexity can be much better than the worst-case, and for the generic avionics mission system task set computational complexity was of no significant concern. The outcome of the schedulability test was that all tasks were completed before their deadlines, a result that was unachievable by synchronous fixed priority scheduling, and asynchronous fixed priority scheduling is suitable for the heavily loaded system.

Dynamically prioritised preemptive scheduling has been analysed and found to successfully schedule the generic avionics mission system tasks. This algorithm is able to utilise up to 100% of processor capacity, but would not provide predictable operation under transient overload, and involves a more complex implementation and higher run-time overheads than fixed priority scheduling.

A statically designed schedule has been demonstrated to successfully schedule the generic avionics mission system tasks. Statically designed schedules are used where deterministic operation is required, or where the run-time overheads of dynamic scheduling can not be accommodated. Disadvantages of static scheduling include the need for significant schedule redesign effort when the timing of any task in the task set changes.

5. References

[1] Douglass C. Locke, David R. Vogel, Lee Lucas and John B. Goodenough, *Generic Avionics Software Specification*, Technical Report CMU/SEI-90-TR-8-ESD-TR-90-209, Software Engineering Institute, Carnegie Mellon University, December 1990.

[2] C. L. Liu and James W. Layland, *Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment*, Journal of the Association for Computing Machinery, Vol. 20, No. 1, pp. 46-61, January 1973.

[3] John Lehoczky, Lui Sha and Ye Ding, 'The Rate Monotonic Scheduling Algorithm: Exact Characterization and Average Case Behavior', in *Proc. Real-Time Systems Symposium (1989)*, IEEE Computer Society Press, pp. 166-171, 1989.

[4] N. C. Audsley, A. Burns, M. F. Richardson and A. J. Wellings, 'Hard Real-Time Scheduling: The Deadline-Monotonic Approach', in *Eighth IEEE Workshop on Real-Time Operating Systems and Software*, pp. 133-137, 1991.

[5] K. Tindell, 'Deadline Monotonic Analysis', *Embedded Systems Programming*, Vol. 13, No. 6 (June), pp. 20-38, 2000.

[6] C. J. Fidge, *Real-Time Scheduling Theory*, Technical Report 02-19, Software Verification Research Centre, University of Queensland, April 2002.

[7] Ken Tindell, *Adding Time-Offsets to Schedulability Analysis*, Technical Report YCS221, Department of Computer Science, University of York, England, 1994.

[8] Joseph Leung and Jennifer Whitehead, *On the Complexity of Fixed-Priority Scheduling of Periodic, Real-Time Tasks*, Performance Evaluation, Vol. 2(4), pp 237-250, Dec 1982.

DISTRIBUTION LIST

An Analysis of Task Scheduling for a Generic Avionics Mission Computer

R. B. Dodd

AUSTRALIA

DEFENCE ORGANISATION	**No. of copies**
Task Sponsor	
Chief Air Operations Division	1 Printed
S&T Program	
Chief Defence Scientist	1
Deputy Chief Defence Scientist Policy	1
AS Science Corporate Management	1
Director General Science Policy Development	1
Counsellor Defence Science, London	Doc Data Sheet
Counsellor Defence Science, Washington	Doc Data Sheet
Scientific Adviser to MRDC, Thailand	Doc Data Sheet
Scientific Adviser Joint	1
Navy Scientific Adviser	Doc Data Sheet & Dist List
Scientific Adviser – Army	Doc Data Sheet & Dist List
Air Force Scientific Adviser	Doc Data Sheet & Dist List
Scientific Adviser to the DMO	Doc Data Sheet & Dist List
Platforms Sciences Laboratory	
Deputy Chief Defence Scientist Aerospace	Doc Data Sht & Exec Summ
Chief of Air Operations Division	Doc Data Sht & Dist List
Research Leader Airborne Mission Systems	1 Printed
Head Airborne Mission Systems Research	1
Head Airborne Mission Systems Operations	1
Head Airborne Mission Systems Networking	1
R. B. Dodd	2 Printed
DSTO Library and Archives	
Library Fishermans Bend	Doc Data Sheet
Library Edinburgh	2 printed
Defence Archives	1 printed
Capability Development Group	
Director General Maritime Development	Doc Data Sheet
Director General Capability and Plans	Doc Data Sheet
Assistant Secretary Investment Analysis	Doc Data Sheet
Director Capability Plans and Programming	Doc Data Sheet
Chief Information Officer Group	
Director General Australian Defence Simulation Office	Doc Data Sheet
AS Information Strategy and Futures	Doc Data Sheet

Director General Information Services	Doc Data Sheet
Strategy Group	
Director General Military Strategy	Doc Data Sheet
Assistant Secretary Governance and Counter-Proliferation	Doc Data Sheet
Navy	
Maritime Operational Analysis Centre, Building 89/90 Garden Island Sydney NSW	Doc Data Sht & Dist List
Deputy Director (Operations)	
Deputy Director (Analysis)	
Director General Navy Capability, Performance and Plans, Navy Headquarters	Doc Data Sheet
Director General Navy Strategic Policy and Futures, Navy Headquarters	Doc Data Sheet
Air Force	
SO (Science) - Headquarters Air Combat Group, RAAF Base, Williamtown NSW 2314	Doc Data Sht & Exec Summary
Army	
ABCA National Standardisation Officer	Doc Data Sheet
Land Warfare Development Sector, Puckapunyal	
SO (Science) - Land Headquarters (LHQ), Victoria Barracks NSW	Doc Data Sht & Exec Summary
SO (Science), Deployable Joint Force Headquarters (DJFHQ) (L), Enoggera QLD	Doc Data Sheet
Joint Operations Command	
Director General Joint Operations	Doc Data Sheet
Chief of Staff Headquarters Joint Operations Command	Doc Data Sheet
Commandant ADF Warfare Centre	Doc Data Sheet
Director General Strategic Logistics	Doc Data Sheet
COS Australian Defence College	Doc Data Sheet
Intelligence and Security Group	
AS Concepts, Capability and Resources	1
DGSTA , Defence Intelligence Organisation	1
Manager, Information Centre, Defence Intelligence Organisation	1
Director Advanced Capabilities	Doc Data Sheet
Defence Materiel Organisation	
Deputy CEO	Doc Data Sheet
Head Aerospace Systems Division	Doc Data Sheet
Head Maritime Systems Division	Doc Data Sheet
Program Manager Air Warfare Destroyer	Doc Data Sheet
CDR Joint Logistics Command	Doc Data Sheet
Guided Weapon & Explosive Ordnance Branch (GWEO)	Doc Data Sheet

OTHER ORGANISATIONS

National Library of Australia	1
NASA (Canberra)	1

UNIVERSITIES AND COLLEGES

Australian Defence Force Academy

Library	1
Head of Aerospace and Mechanical Engineering	1
Hargrave Library, Monash University	Doc Data Sheet

OUTSIDE AUSTRALIA

INTERNATIONAL DEFENCE INFORMATION CENTRES

US Defense Technical Information Center	1
UK Dstl Knowledge Services	1
Canada Defence Research Directorate R&D Knowledge & Information Management (DRDKIM)	1
NZ Defence Information Centre	1

ABSTRACTING AND INFORMATION ORGANISATIONS

Library, Chemical Abstracts Reference Service	1
Engineering Societies Library, US	1
Materials Information, Cambridge Scientific Abstracts, US	1
Documents Librarian, The Center for Research Libraries, US	1

SPARES	5 Printed

Total number of copies: 35 **Printed:** 12 **PDF:** 23

Page classification: UNCLASSIFIED

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION
DOCUMENT CONTROL DATA

1. PRIVACY MARKING/CAVEAT (OF DOCUMENT)

2. TITLE

An Analysis of Task Scheduling for a Generic Avionics Mission Computer

3. SECURITY CLASSIFICATION (FOR UNCLASSIFIED REPORTS THAT ARE LIMITED RELEASE USE (L) NEXT TO DOCUMENT CLASSIFICATION)

Document	(U)
Title	(U)
Abstract	(U)

4. AUTHOR(S)

R. B. Dodd

5. CORPORATE AUTHOR

DSTO Defence Science and Technology Organisation
506 Lorimer St
Fishermans Bend Victoria 3207 Australia

6a. DSTO NUMBER	6b. AR NUMBER	6c. TYPE OF REPORT	7. DOCUMENT DATE
DSTO-TN-0691	AR-013-635	Technical Note	April 2006

8. FILE NUMBER	9. TASK NUMBER	10. TASK SPONSOR	11. NO. OF PAGES	12. NO. OF REFERENCES
E9505-25-70	DST 00/061	CAOD	29	8

13. URL on the World Wide Web

http://www.dsto.defence.gov.au/corporate/reports/DSTO-TN-0691.pdf

14. RELEASE AUTHORITY

Chief, Air Operations Division

15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT

Approved for public release

OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE, PO BOX 1500, EDINBURGH, SA 5111

16. DELIBERATE ANNOUNCEMENT

No Limitations

17. CITATION IN OTHER DOCUMENTS Yes

18. DSTO Research Library Thesaurus

Aircraft systems, Scheduling, Algorithms.

19. ABSTRACT

Task scheduling is investigated for a set of generic tasks representative of a heavily loaded mission computer application. A number of commonly used scheduling algorithms are applied to the generic task set, and a range of schedulability analysis calculations are performed. Suitability of the scheduling algorithms to the mission computer application is determined, and characteristics of different schedulability analysis techniques are established through their application to the generic task set.